LISTA Tutorial – Improved Synthesis Models

Minimum Entropy Rate Simplification and Multiplicative-Mixture HMMs

Gustav Eje Henter & W. Bastiaan Kleijn

Sound and Image Processing Laboratory
School of Electrical Engineering
KTH – Royal Institute of Technology
Stockholm, Sweden

3rd September 2010

Presentation Outline

1. Introduction
2. Previous work: Minimum entropy rate simplification (MERS)
    1. Theory
    2. Examples
3. Ongoing work:
    1. MERS for HMMs
    2. Multiplicative-mixture HMMs

Presentation Outline

1. Introduction
2. Previous work: Minimum entropy rate simplification (MERS)
    1. Theory
    2. Examples
3. Ongoing work:
    1. MERS for HMMs
    2. Multiplicative-mixture HMMs

A Fool's Approach to Speech Synthesis

1. Acquire speech data.
2. Train generative model on data.
3. Sample from speech model.
4. PROFIT! (You now have speech.)

This scheme falls apart at step 3. Sampling from the model does not actually produce speech!
Traditional models for speech synthesis are poor representations of human speech in general. They are most accurate at the mode; this is the basis for maximum likelihood parameter generation.

A Fool's Approach to Speech Synthesis

1. Acquire speech data.
2. Train generative model on data.
3. Sample from speech model.
4. PROFIT! (You now have speech.)

This scheme falls apart at step 3. Sampling from the model does not actually produce speech!
Traditional models for speech synthesis are poor representations of human speech in general. They are most accurate at the mode; this is the basis for maximum likelihood parameter generation.

Unifying Theme

The unifying theme of this presentation is proposals for improving models used in synthesis.

1. Minimum entropy rate simplification. This improves models for sampling by cutting away the tails, typically the worst fitting parts of the model.
2. Multiplicative-mixture HMMs. These can provide better models of gradually changing processes such as speech.

Presentation Outline

1. Introduction
2. Previous work: Minimum entropy rate simplification (MERS)
    1. Theory
    2. Examples
3. Ongoing work:
    1. MERS for HMMs
    2. Multiplicative-mixture HMMs

Minimum Entropy Rate Simplification

Recent work at KTH (EUSIPCO 2010, second paper awaiting submission).

- A model improvement scheme
  - Denoising, simplifying, or removing poorly fitting parts of models.
- Related to rate-distortion theory
  - Representation-independent and based on information theory.
  - Enables re-use of established tools and ideas.
- Improves the quality of samples from a model

MERS Use Cases

MERS was developed for models in generative settings:

- Models trained on data with interference, e.g., field recordings of speech or birdsong
  - It is desirable to remove interferences from the model
  - We may not have a model of the disturbances
  - Background interference will be part of description even if $N \to \infty$
  - Post-processing the model scales better than denoising the data itself
  - Bayesian models require a prior, which diminishes in importance as $N \to \infty$
- Model simplification
- Model fits data poorly
  - Reduce the range of behaviours of the model

Generative vs. Discriminative Tasks

There are two kinds of models in machine learning—discriminative models for $P(Y \mid X)$, and generative models describing $P(Y, X)$.
Not every type is appropriate for every task:

- Discriminative tasks, like classification, map $X \to Y$ (data to class) using $P(Y \mid X)$. These can be solved by either model type.
- Synthesis tasks or sampling go from $Y \to X$ (class to data) using the joint distribution $P(Y, X)$. These require generative models.

Model Post-Processing

Generative models can be used with the same training algorithms both to discriminate and synthesise data, but each calls for a different approach:

- In recognition tasks, it is common to post-process models to improve robustness. Smoothing may for example be applied to increase the variability of models. This compensates for how the real world usually shows more variation and exhibits more phenomena than the training data; assigning zero probability to unseen events, like ML typically does, would prevent recognition.
- In a generative scenario, there may be reasons to reduce variation instead. Removing or de-emphasising uncommon outcomes, in particular, makes the output less random and more characteristic. We call this *probability concentration*, similar to energy concentration in coding. We soon present additional reasons why this is a good idea.

Probability concentration is related to sparsity, but it is not the same. Sparsity is representation-dependent as it applies to coefficients in a specific basis.

Model Post-Processing

Generative models can be used with the same training algorithms both to discriminate and synthesise data, but each calls for a different approach:

- In recognition tasks, it is common to post-process models to improve robustness. Smoothing may for example be applied to increase the variability of models. This compensates for how the real world usually shows more variation and exhibits more phenomena than the training data; assigning zero probability to unseen events, like ML typically does, would prevent recognition.
- In a generative scenario, there may be reasons to reduce variation instead. Removing or de-emphasising uncommon outcomes, in particular, makes the output less random and more characteristic. We call this *probability concentration*, similar to energy concentration in coding. We soon present additional reasons why this is a good idea.

Probability concentration is related to sparsity, but it is not the same. Sparsity is representation-dependent as it applies to coefficients in a specific basis.

Model Post-Processing

Generative models can be used with the same training algorithms both to discriminate and synthesise data, but each calls for a different approach:

- In recognition tasks, it is common to post-process models to improve robustness. Smoothing may for example be applied to increase the variability of models. This compensates for how the real world usually shows more variation and exhibits more phenomena than the training data; assigning zero probability to unseen events, like ML typically does, would prevent recognition.
- In a generative scenario, there may be reasons to reduce variation instead. Removing or de-emphasising uncommon outcomes, in particular, makes the output less random and more characteristic. We call this *probability concentration*, similar to energy concentration in coding. We soon present additional reasons why this is a good idea.

Probability concentration is related to sparsity, but it is not the same. Sparsity is representation-dependent as it applies to coefficients in a specific basis.

Presentation Outline

1. Introduction
2. Previous work: Minimum entropy rate simplification (MERS)
    1. Theory
    2. Examples
3. Ongoing work:
    1. MERS for HMMs
    2. Multiplicative-mixture HMMs

Notation

Given: A stationary, ergodic stochastic process model $\widetilde{X} = \left\{\widetilde{X}_t : t \in \mathbb{Z}\right\}$. This may have been learned from disturbed data, or there are otherwise reasons to simplify it.

Given: A class of stationary, ergodic stochastic processes $\mathcal{X}$. Typically $\widetilde{X} \in \mathcal{X}$.

Desired: A simplified or denoised model $X = \{X_t : t \in \mathbb{Z}\} \in \mathcal{X}$.

Notation: $X^\star \in \mathcal{X}$ represents the underlying, undisturbed process to identify, or its closest approximation in $\mathcal{X}$.

Omitted: Noise models of any sort. We want to apply a non-parametric principle.

Key Assumption

Key assumption: *Uncommon outcomes are more likely to be noise.* (X is thus simpler than $\widetilde{X}$.)

- Removing uncommon behaviours then increases model SNR.
- Results depend on signal-noise separation and noise probability magnitudes.

The improved SNR needs to be balanced against the degree of simplification that is acceptable in any particular scenario.

Inspiration in Rate-Distortion Theory

To find the optimally simplified X given constraints on the distance from $\widetilde{X}$, we look to *rate-distortion theory*, a constrained optimisation framework from source coding. There are many reasons for this:

1. The goal of rate-distortion theory is to produce optimally simplified descriptions (specifically for compression).
2. It is representation-independent, as it originates in information theory.
3. Well-known solutions exhibit reverse water-filling, a kind of sparsity. We would like to similarly enable probability concentration.
4. It has already spawned several machine learning spin-offs, such as information bottleneck.
5. We may re-use tools and solutions from an already well developed field.

Inspiration in Rate-Distortion Theory

To find the optimally simplified X given constraints on the distance from $\widetilde{X}$, we look to *rate-distortion theory*, a constrained optimisation framework from source coding. There are many reasons for this:

1. The goal of rate-distortion theory is to produce optimally simplified descriptions (specifically for compression).
2. It is representation-independent, as it originates in information theory.
3. Well-known solutions exhibit reverse water-filling, a kind of sparsity. We would like to similarly enable probability concentration.
4. It has already spawned several machine learning spin-offs, such as information bottleneck.
5. We may re-use tools and solutions from an already well developed field.

Inspiration in Rate-Distortion Theory

To find the optimally simplified X given constraints on the distance from $\widetilde{X}$, we look to *rate-distortion theory*, a constrained optimisation framework from source coding. There are many reasons for this:

1. The goal of rate-distortion theory is to produce optimally simplified descriptions (specifically for compression).
2. It is representation-independent, as it originates in information theory.
3. Well-known solutions exhibit reverse water-filling, a kind of sparsity. We would like to similarly enable probability concentration.
4. It has already spawned several machine learning spin-offs, such as information bottleneck.
5. We may re-use tools and solutions from an already well developed field.

Inspiration in Rate-Distortion Theory

To find the optimally simplified X given constraints on the distance from $\widetilde{X}$, we look to *rate-distortion theory*, a constrained optimisation framework from source coding. There are many reasons for this:

1. The goal of rate-distortion theory is to produce optimally simplified descriptions (specifically for compression).
2. It is representation-independent, as it originates in information theory.
3. Well-known solutions exhibit reverse water-filling, a kind of sparsity. We would like to similarly enable probability concentration.
4. It has already spawned several machine learning spin-offs, such as information bottleneck.
5. We may re-use tools and solutions from an already well developed field.

Inspiration in Rate-Distortion Theory

To find the optimally simplified X given constraints on the distance from $\widetilde{X}$, we look to *rate-distortion theory*, a constrained optimisation framework from source coding. There are many reasons for this:

1. The goal of rate-distortion theory is to produce optimally simplified descriptions (specifically for compression).
2. It is representation-independent, as it originates in information theory.
3. Well-known solutions exhibit reverse water-filling, a kind of sparsity. We would like to similarly enable probability concentration.
4. It has already spawned several machine learning spin-offs, such as information bottleneck.
5. We may re-use tools and solutions from an already well developed field.

Inspiration in Rate-Distortion Theory

To find the optimally simplified X given constraints on the distance from $\widetilde{X}$, we look to *rate-distortion theory*, a constrained optimisation framework from source coding. There are many reasons for this:

1. The goal of rate-distortion theory is to produce optimally simplified descriptions (specifically for compression).
2. It is representation-independent, as it originates in information theory.
3. Well-known solutions exhibit reverse water-filling, a kind of sparsity. We would like to similarly enable probability concentration.
4. It has already spawned several machine learning spin-offs, such as information bottleneck.
5. We may re-use tools and solutions from an already well developed field.

Quantifying Simplicity

How do we quantify simplicity—the analogue of rate? And how to do this so that probability concentration (sparsity) is encouraged?

A common information-theoretic complexity measure is *information entropy*

$$H(Y) = -\sum_i p_Y(i) \log p_Y(i) \geq 0.$$

This is representation-independent and relates to how hard a variable is to compress, the bitrate. For continuous-valued variables we use an integral. For stochastic processes, one usually considers the *entropy rate*

$$H_\infty(X) = \lim_{T \to \infty} \frac{1}{T} H(\{X_{t+1}, \ldots, X_{t+T}\}).$$

Minimising entropy rate should produce simpler processes. Minimal entropy rate processes are deterministic.

Quantifying Simplicity

How do we quantify simplicity—the analogue of rate? And how to do this so that probability concentration (sparsity) is encouraged?
A common information-theoretic complexity measure is *information entropy*

$$H(Y) = -\sum_i p_Y(i) \log p_Y(i) \geq 0.$$

This is representation-independent and relates to how hard a variable is to compress, the bitrate. For continuous-valued variables we use an integral.
For stochastic processes, one usually considers the *entropy rate*

$$H_\infty(X) = \lim_{T\to\infty} \frac{1}{T} H(\{X_{t+1}, \ldots, X_{t+T}\}).$$

Minimising entropy rate should produce simpler processes. Minimal entropy rate processes are deterministic.

Quantifying Simplicity

How do we quantify simplicity—the analogue of rate? And how to do this so that probability concentration (sparsity) is encouraged?
A common information-theoretic complexity measure is *information entropy*

$$H(Y) = -\sum_i p_Y(i) \log p_Y(i) \geq 0.$$

This is representation-independent and relates to how hard a variable is to compress, the bitrate. For continuous-valued variables we use an integral. For stochastic processes, one usually considers the *entropy rate*

$$H_\infty(X) = \lim_{T\to\infty} \frac{1}{T} H(\{X_{t+1}, \ldots, X_{t+T}\}).$$

Minimising entropy rate should produce simpler processes. Minimal entropy rate processes are deterministic.

Preventing Oversimplification

Rate minimisation in rate-distortion theory is always subject to a distortion constraint, restricting compression errors. We similarly constrain X not to stray too far from the original $\widetilde{X}$. How do we quantify this dissimilarity? And how to do this so that probability concentration is encouraged?

A parameterisation-independent similarity measure is *Kullback-Leibler divergence* or *relative entropy*

$$D_{\mathrm{KL}}(P \,||\, Q) = \sum_i p_P(i) \log \frac{p_P(i)}{p_Q(i)} \geq 0.$$

Like entropy, this is measured in bits. Again we may use integrals for continuous-valued P and Q.

For stationary stochastic processes, an analogous *relative entropy rate* is

$$D_\infty(P \,||\, Q) = \lim_{T\to\infty} \frac{1}{T} D_{\mathrm{KL}}\left(\{P_{t+1}, \ldots, P_{t+T}\} \,||\, \{Q_{t+1}, \ldots, Q_{t+T}\}\right)$$

Preventing Oversimplification

Rate minimisation in rate-distortion theory is always subject to a distortion constraint, restricting compression errors. We similarly constrain X not to stray too far from the original $\widetilde{X}$. How do we quantify this dissimilarity? And how to do this so that probability concentration is encouraged?
A parameterisation-independent similarity measure is *Kullback-Leibler divergence* or *relative entropy*

$$D_{\mathrm{KL}}(P \,||\, Q) = \sum_i p_P(i) \log \frac{p_P(i)}{p_Q(i)} \geq 0.$$

Like entropy, this is measured in bits. Again we may use integrals for continuous-valued P and Q.
For stationary stochastic processes, an analogous *relative entropy rate* is

$$D_\infty(P \,||\, Q) = \lim_{T\to\infty} \frac{1}{T} D_{\mathrm{KL}}\left(\{P_{t+1}, \ldots, P_{t+T}\} \,||\, \{Q_{t+1}, \ldots, Q_{t+T}\}\right)$$

Allowing Probability Concentration

In the KL-divergence, P is the true data distribution (typically assumed known in coding), while Q is an approximation to be optimised. The divergence is highly averse to excess sparsity in Q—$p_P(i) > 0$ while $p_Q(i) = 0$ generates infinite divergence. In a coding interpretation this implies an infinitely long codeword for P.

Our proposed situation is different: we know only an approximation $\widetilde{X}$ of the true distribution X we are interested in. It makes sense to constrain

$$D_\infty(X \,||\, \widetilde{X}) \leq D$$

and seek X. We thus fix argument "Q" rather than "P" and optimise, similar to variational Bayes methods.

This measure is very averse to adding behaviours to X that are not in $\widetilde{X}$, but only gives finite penalty if behaviours are taken away. This should promote rather than punish probability concentration.

Allowing Probability Concentration

In the KL-divergence, P is the true data distribution (typically assumed known in coding), while Q is an approximation to be optimised. The divergence is highly averse to excess sparsity in Q—$p_P(i) > 0$ while $p_Q(i) = 0$ generates infinite divergence. In a coding interpretation this implies an infinitely long codeword for P.
Our proposed situation is different: we know only an approximation $\widetilde{X}$ of the true distribution X we are interested in. It makes sense to constrain

$$D_\infty(X \,||\, \widetilde{X}) \leq D$$

and seek X. We thus fix argument "Q" rather than "P" and optimise, similar to variational Bayes methods.
This measure is very averse to adding behaviours to X that are not in $\widetilde{X}$, but only gives finite penalty if behaviours are taken away. This should promote rather than punish probability concentration.

Allowing Probability Concentration

In the KL-divergence, P is the true data distribution (typically assumed known in coding), while Q is an approximation to be optimised. The divergence is highly averse to excess sparsity in Q—$p_P(i) > 0$ while $p_Q(i) = 0$ generates infinite divergence. In a coding interpretation this implies an infinitely long codeword for P.
Our proposed situation is different: we know only an approximation $\widetilde{X}$ of the true distribution X we are interested in. It makes sense to constrain

$$D_\infty(X \,||\, \widetilde{X}) \leq D$$

and seek X. We thus fix argument "Q" rather than "P" and optimise, similar to variational Bayes methods.
This measure is very averse to adding behaviours to X that are not in $\widetilde{X}$, but only gives finite penalty if behaviours are taken away. This should promote rather than punish probability concentration.

General Minimum Entropy Rate Simplification

We can now formulate the general *minimum entropy rate simplification* (MERS) framework:
Given a stochastic process $\widetilde{X}$, a class of stochastic processes $\mathcal{X}$, and a divergence bound D, the minimum entropy rate simplification X of $\widetilde{X}$ in $\mathcal{X}$ solves

$$\min_{X \in \mathcal{X}} H_\infty(X)$$

subject to

$$D_\infty(X \,||\, \widetilde{X}) \leq D,$$

assuming these quantities exist.

Remarks on General MERS

- We think of X as a simple process for generating the observations, minus many random corruptions and disturbances.
- The optimisation is very similar to rate-distortion theory, as desired.
- The proposal extends naturally to continuous-valued processes.
- Because we consider limiting quantities for long sequences of contiguous samples, we are operating over and simplifying entire behaviours, rather than mere outcomes or samples.
- The problem is nonconvex.

Presentation Outline

1. Introduction
2. Previous work: Minimum entropy rate simplification (MERS)
    1. Theory
    2. Examples
3. Ongoing work:
    1. MERS for HMMs
    2. Multiplicative-mixture HMMs

Example I: Gaussian Processes

What does this imply in practise?

As a first example, let $\mathcal{X}$ be the space of purely nondeterministic stationary, ergodic, univariate Gaussian processes, and let $\widetilde{X} \in \mathcal{X}$. Define the spectral density functions of $\widetilde{X}$ and X to be $R_{\widetilde{X}}(\omega)$ and $R_X(\omega)$.

The rate is related to $\log R_X(\omega)$ integrated over ω, while the divergence becomes the Itakura-Saito divergence. The MERS problem can be solved using a variational approach, using a Lagrange multiplier $\lambda > 0$ for the divergence constraint. The solution, for $\lambda > 1$, is

$$R_X(\omega) = \frac{\lambda - 1}{\lambda} R_{\widetilde{X}}(\omega).$$

Example I: Gaussian Processes

What does this imply in practise?
As a first example, let $\mathcal{X}$ be the space of purely nondeterministic stationary, ergodic, univariate Gaussian processes, and let $\widetilde{X} \in \mathcal{X}$. Define the spectral density functions of $\widetilde{X}$ and X to be $R_{\widetilde{X}}(\omega)$ and $R_X(\omega)$.
The rate is related to $\log R_X(\omega)$ integrated over ω, while the divergence becomes the Itakura-Saito divergence. The MERS problem can be solved using a variational approach, using a Lagrange multiplier $\lambda > 0$ for the divergence constraint. The solution, for $\lambda > 1$, is

$$R_X(\omega) = \frac{\lambda - 1}{\lambda} R_{\widetilde{X}}(\omega).$$

Remarks on the Solution

We can define $\alpha = \frac{\lambda}{\lambda - 1} > 1$. The solution then corresponds to multiplying the variance of the driving Gaussian noise by α^{-1}, simple variance scaling. We can also write

$$f_{X_t|\underline{X}_{t-T}^{t-1}}(x_t \mid \underline{x}_{t-T}^{t-1}) = \frac{1}{\nu}\left(f_{\widetilde{X}_t|\underline{\widetilde{X}}_{t-T}^{t-1}}(x_t \mid \underline{x}_{t-T}^{t-1})\right)^{\alpha}.$$

Since the ratio between pdfs $f_{X_t|\underline{X}_{t-T}^{t-1}}$ and $f_{\widetilde{X}_t|\underline{\widetilde{X}}_{t-T}^{t-1}}$ is strictly increasing in $f_{\widetilde{X}_t|\underline{\widetilde{X}}_{t-T}^{t-1}}$, we have probability concentration.

Adding a Variance Constraint

A more interesting solution is obtained if we constrain $\mathcal{X}$ to be the set of processes with the same variance as $\widetilde{X}$, to prevent shrinking. (This also means the minimum rate solution cannot be completely deterministic.)

A similar variational optimisation now yields

$$R_X(\omega) = \frac{1}{\nu} \frac{R_{\widetilde{X}}(\omega)}{R_{\widetilde{X}}(\omega) + \frac{\alpha}{\nu}}$$

as the MERS solution for a set of Lagrange multiplier values α, ν. This can be shown to erode away valleys in the spectrum while peaks are emphasised, akin to reverse water-filling.

Adding a Variance Constraint

A more interesting solution is obtained if we constrain $\mathcal{X}$ to be the set of processes with the same variance as $\widetilde{X}$, to prevent shrinking. (This also means the minimum rate solution cannot be completely deterministic.)
A similar variational optimisation now yields

$$R_X(\omega) = \frac{1}{\nu} \frac{R_{\widetilde{X}}(\omega)}{R_{\widetilde{X}}(\omega) + \frac{\alpha}{\nu}}$$

as the MERS solution for a set of Lagrange multiplier values α, ν. This can be shown to erode away valleys in the spectrum while peaks are emphasised, akin to reverse water-filling.

General Gaussian Processes

If we consider a general Gaussian process $\widetilde{X}'$, this can be written as $\widetilde{X}' = \widetilde{\mu} + \widetilde{X}$ using the Wold decomposition, where $\widetilde{\mu}$ is purely deterministic and $\widetilde{X}$ purely nondeterministic as before.
The MERS solution is then $X' = \widetilde{\mu} + X$, i.e., simplify $\widetilde{X}$ as before, but leave the deterministic part unchanged. (We here assume $\mathcal{X}$ is closed under deterministic additions.)

Example II: Markov Chains

Our second MERS example considers first-order, discrete Markov chains

$$P\left(\widetilde{X}_{t+1} \mid \widetilde{X}_t, \widetilde{X}_{t-1}, \ldots\right) = P\left(\widetilde{X}_{t+1} \mid \widetilde{X}_t\right).$$

Any Markov chain of finite order can be reduced to this case. No additional constraints are necessary, since there is no concept of a variance that can be shrunk.

Preliminaries

We let $\widetilde{X}$ be a Markov chain parametrised by the transition matrix $\widetilde{\boldsymbol{A}}$ with elements $\widetilde{a}_{ij} = P\left(\widetilde{X}_{t+1} = j \mid \widetilde{X}_t = i\right)$, and similarly with $\boldsymbol{A}$ for X. We also define the stationary distribution vector $\boldsymbol{\pi}$ such that $\pi_i = P(X_t = i)$, and require $\boldsymbol{\pi} > \mathbf{0}$. $\boldsymbol{A}$ must satisfy $\boldsymbol{A}\mathbf{1} = \mathbf{1}$ and $\boldsymbol{A} \geq \mathbf{0}$ to describe a valid Markov chain.

The Markov Chain MERS Problem

The entropy rate for a general Markov chain is

$$H_\infty(X) = -\sum_i \pi_i \sum_j a_{ij} \log a_{ij},$$

and the MERS-relevant KL-divergence is similarly

$$D_\infty(X \,||\, \widetilde{X}) = \sum_i \pi_i \sum_j a_{ij} \log \frac{a_{ij}}{\widetilde{a}_{ij}}.$$

This nonconvex problem is complicated to solve, particularly because expressions involve the stationary distribution π, the leading eigenvector of $\boldsymbol{A}^T$ and a highly complex function of the parameters a_{ij}. Numerical solutions are slow and often fall into local minima.

The Markov Chain MERS Problem

The entropy rate for a general Markov chain is

$$H_\infty(X) = -\sum_i \pi_i \sum_j a_{ij} \log a_{ij},$$

and the MERS-relevant KL-divergence is similarly

$$D_\infty(X \,||\, \widetilde{X}) = \sum_i \pi_i \sum_j a_{ij} \log \frac{a_{ij}}{\widetilde{a}_{ij}}.$$

This nonconvex problem is complicated to solve, particularly because expressions involve the stationary distribution $\boldsymbol{\pi}$, the leading eigenvector of $\boldsymbol{A}^T$ and a highly complex function of the parameters a_{ij}. Numerical solutions are slow and often fall into local minima.

A Bigram Formulation

A better idea is to parameterise the problem using the *bigram probability matrix* $\mathbf{B}$ with elements $b_{ij} = P(X_t = i \cap X_{t+1} = j)$. This satisfies $\boldsymbol{\pi} = \boldsymbol{B}\mathbf{1}$, and we have the conversion formula $\boldsymbol{B} = \operatorname{diag}(\boldsymbol{\pi})\boldsymbol{A}$.

In this parameterisation, the MERS problem can be expressed without eigenvectors as

$$\min_{\boldsymbol{B}} -\sum_{ij} b_{ij} \log \frac{b_{ij}}{\sum_{j'} b_{ij'}}$$

subject to

$$\sum_{ij} b_{ij} \log \frac{b_{ij}}{\widetilde{a}_{ij} \sum_{j'} b_{ij'}} \leq D$$

$$B\mathbf{1} - B^T\mathbf{1} = 0$$

$$\mathbf{1}^T B\mathbf{1} = 1$$

$$B \geq 0.$$

A Bigram Formulation

A better idea is to parameterise the problem using the *bigram probability matrix* $\mathbf{B}$ with elements $b_{ij} = P(X_t = i \cap X_{t+1} = j)$. This satisfies $\boldsymbol{\pi} = \boldsymbol{B}\mathbf{1}$, and we have the conversion formula $\boldsymbol{B} = \operatorname{diag}(\boldsymbol{\pi})\,\boldsymbol{A}$.
In this parameterisation, the MERS problem can be expressed without eigenvectors as

$$\min_{\boldsymbol{B}} -\sum_{ij} b_{ij} \log \frac{b_{ij}}{\sum_{j'} b_{ij'}}$$

subject to

$$\sum_{ij} b_{ij} \log \frac{b_{ij}}{\widetilde{a}_{ij} \sum_{j'} b_{ij'}} \leq D$$

$$\boldsymbol{B}\mathbf{1} - \boldsymbol{B}^T\mathbf{1} = \mathbf{0}$$

$$\mathbf{1}^T\boldsymbol{B}\mathbf{1} = 1$$

$$\boldsymbol{B} \geq \mathbf{0}.$$

MERS Solution for Markov Chains

We can use the trick from the Blahut-Arimoto algorithm from rate-distortion theory to derive an iterative solution procedure for this problem. Surprisingly, the fixed point of the iterations can then be found analytically. This gives the globally optimal MERS solution as

$$\boldsymbol{A} = \frac{1}{\nu} \left(\operatorname{diag} \boldsymbol{\mu}\right)^{-1} \widetilde{\boldsymbol{A}}^{\cdot\alpha} \left(\operatorname{diag} \boldsymbol{\mu}\right),$$

where $\widetilde{\boldsymbol{A}}^{\cdot\alpha}$ denotes elementwise exponentiation of $\widetilde{\boldsymbol{A}}$ by α, while $\boldsymbol{\mu}$ is the positive leading right eigenvector of $\widetilde{\boldsymbol{A}}^{\cdot\alpha}$ with eigenvalue $\nu > 0$. Thus $(\widetilde{\boldsymbol{A}}^{\cdot\alpha})_{ij} = (\widetilde{a}_{ij})^{\alpha}$ and $\widetilde{\boldsymbol{A}}^{\cdot\alpha}\boldsymbol{\mu} = \nu\boldsymbol{\mu}$.
The exponent $\alpha = \frac{\lambda}{\lambda-1} > 1$ is defined through the Lagrange multiplier λ. Fixing α corresponds to a particular simplicity-divergence trade-off.

Remarks on the Solution

It is easy to verify that the optimal $\boldsymbol{A}$ is a valid transition matrix. The sandwich operation using the two eigenvector matrices is another kind of matrix normalisation, compared to $\operatorname{diag}(\boldsymbol{C}\mathbf{1})^{-1}\boldsymbol{C}$ used in maximum likelihood estimation. It is similar to expressions that arise in connection with time-reversible Markov chains.

In the solutions, transitions are never added or entirely removed, but may become arbitrarily unlikely due to the erosion from α. We can express the next-step pdf as

$$p_{X_t|\underline{X}_{t-T}^{t-1}}(x_t \mid \underline{x}_{t-T}^{t-1}) = \frac{1}{\nu}\frac{\mu_{x_{t-1}}}{\mu_{x_t}}\left(p_{\widetilde{X}_t|\underline{\widetilde{X}}_{t-T}^{t-1}}(x_t \mid \underline{x}_{t-T}^{t-1})\right)^{\alpha}.$$

The only difference from the Gaussian case is the normalisation due to μ, which ensuring that the solution satisfies $\boldsymbol{A}\mathbf{1} = \mathbf{1}$ as required.

Remarks on the Solution

It is easy to verify that the optimal $\boldsymbol{A}$ is a valid transition matrix. The sandwich operation using the two eigenvector matrices is another kind of matrix normalisation, compared to $\operatorname{diag}(\boldsymbol{C}\mathbf{1})^{-1}\boldsymbol{C}$ used in maximum likelihood estimation. It is similar to expressions that arise in connection with time-reversible Markov chains.
In the solutions, transitions are never added or entirely removed, but may become arbitrarily unlikely due to the erosion from α. We can express the next-step pdf as

$$p_{X_t|\underline{X}_{t-T}^{t-1}}(x_t \mid \underline{x}_{t-T}^{t-1}) = \frac{1}{\nu}\frac{\mu_{x_{t-1}}}{\mu_{x_t}}\left(p_{\widetilde{X}_t|\underline{\widetilde{X}}_{t-T}^{t-1}}(x_t \mid \underline{x}_{t-T}^{t-1})\right)^{\alpha}.$$

The only difference from the Gaussian case is the normalisation due to $\boldsymbol{\mu}$, which ensuring that the solution satisfies $\boldsymbol{A}\mathbf{1} = \mathbf{1}$ as required.

Numerical Example

For a numerical example, we fit a second-order Markov grammar $\widetilde{X}$ to a model of spontaneous speech as concatenated random draws from a bag of sentences, corrupted by errors characteristic of spontaneous speech. We also fit a reference grammar $X^\star$ to the same bag of sentences without errors.

The sentences contained 21 word tokens and a pause marker. We then got a 458×458 $\widetilde{A}$-matrix having 5550 nonzero elements. In contrast, $B^\star$ was very sparse, with only 135 nonzero probabilities.

We applied MERS to $\widetilde{X}$ to obtain a simplified and denoised model X.

Method performance and solution properties are illustrated in the plots on the next few slides.

Numerical Example

For a numerical example, we fit a second-order Markov grammar $\widetilde{X}$ to a model of spontaneous speech as concatenated random draws from a bag of sentences, corrupted by errors characteristic of spontaneous speech. We also fit a reference grammar $X^\star$ to the same bag of sentences without errors.

The sentences contained 21 word tokens and a pause marker. We then got a 458×458 $\widetilde{\boldsymbol{A}}$-matrix having 5550 nonzero elements. In contrast, $\boldsymbol{B}^\star$ was very sparse, with only 135 nonzero probabilities.

We applied MERS to $\widetilde{X}$ to obtain a simplified and denoised model X. Method performance and solution properties are illustrated in the plots on the next few slides.

Numerical Example

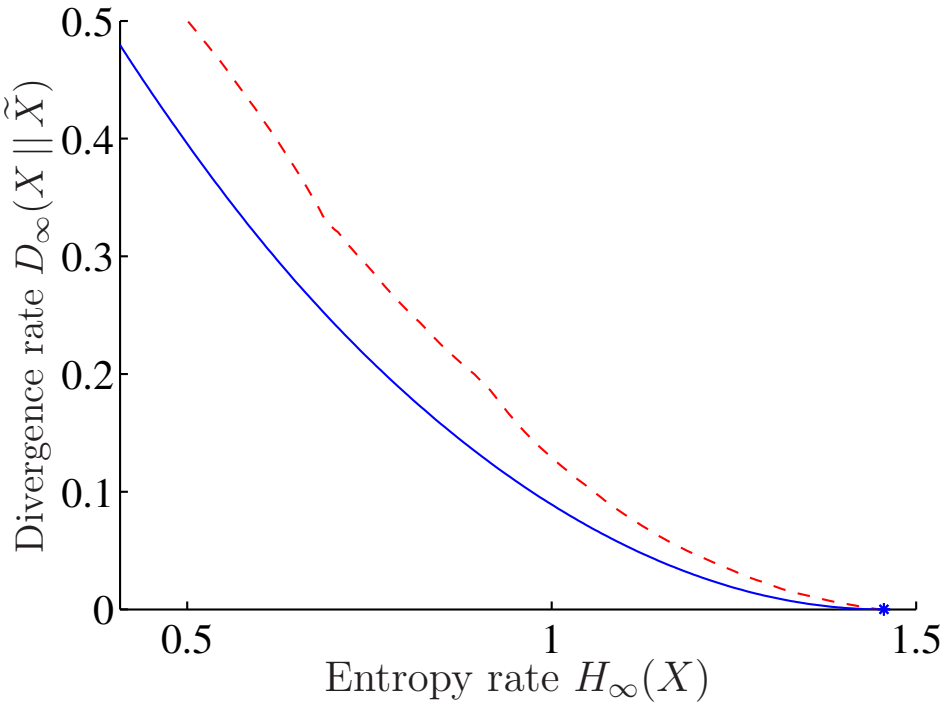
For a numerical example, we fit a second-order Markov grammar $\widetilde{X}$ to a model of spontaneous speech as concatenated random draws from a bag of sentences, corrupted by errors characteristic of spontaneous speech. We also fit a reference grammar $X^\star$ to the same bag of sentences without errors.

The sentences contained 21 word tokens and a pause marker. We then got a 458×458 $\widetilde{\boldsymbol{A}}$-matrix having 5550 nonzero elements. In contrast, $\boldsymbol{B}^\star$ was very sparse, with only 135 nonzero probabilities.

We applied MERS to $\widetilde{X}$ to obtain a simplified and denoised model X.

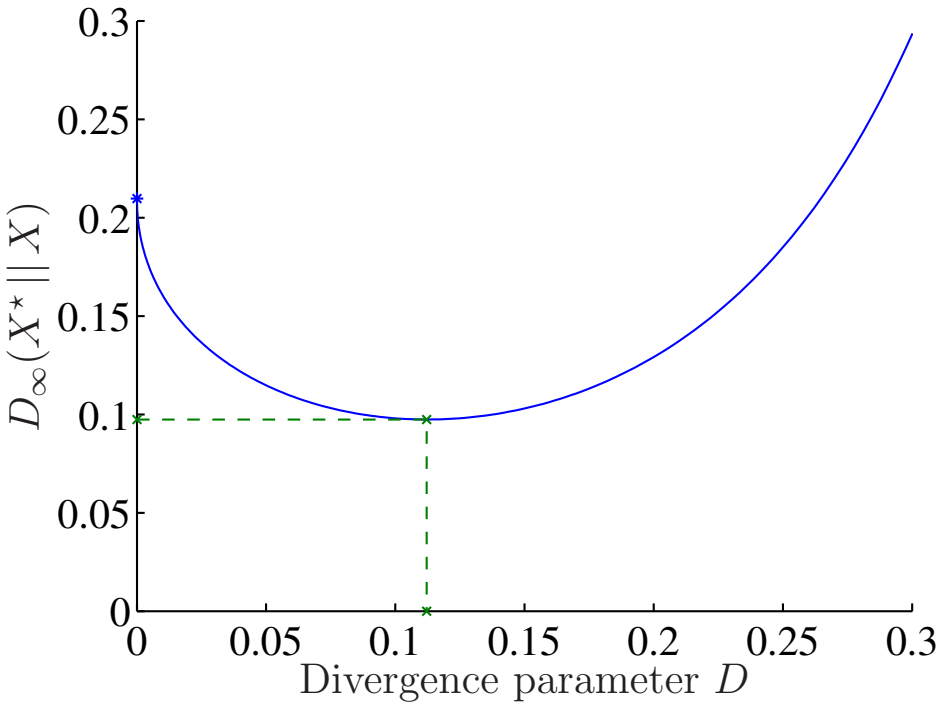
Method performance and solution properties are illustrated in the plots on the next few slides.

Rate-Divergence Performance



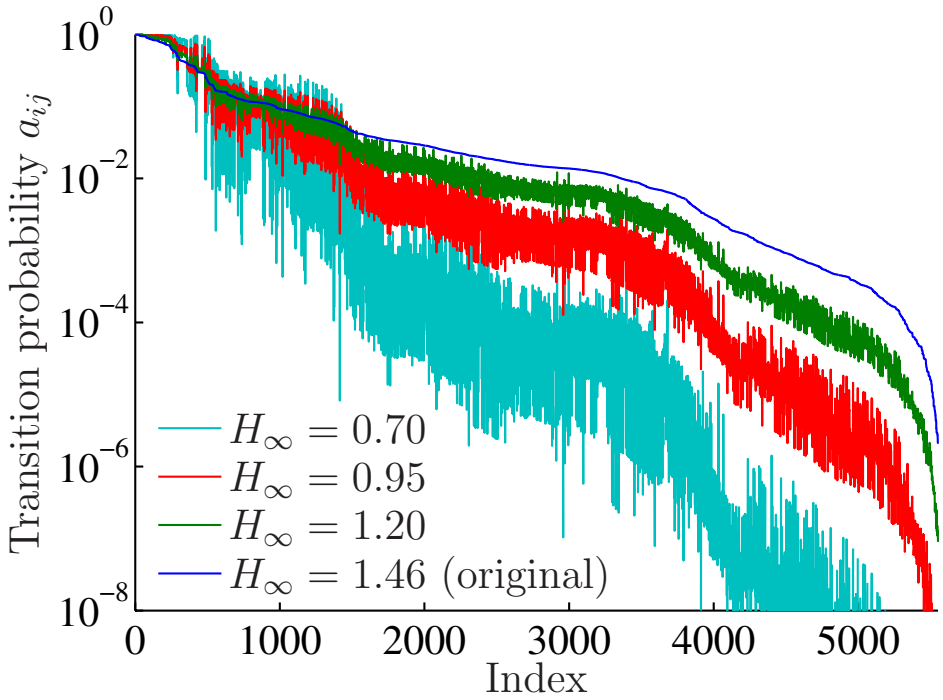
The dashed curve is a simple scheme thresholding the rows of $\widetilde{\boldsymbol{A}}$, similar to reverse water-filling.

Denoising Performance



Similar to what humans do, a much better (though not perfect) grammar is recovered from corrupted data alone.

Transition Probabilities



The lower the rate of the solution, the more $\boldsymbol{A}$ concentrates on a few selected behaviours. This is sparse since most probabilities rapidly become very small. The noisy appearance is due to the normalising $\boldsymbol{\mu}$.

Summary

- MERS takes a stochastic process model (e.g., a Gaussian process or a Markov chain) and denoises, simplifies, or concentrates it on its mode by de-emphasising uncommon behaviours.
- The degree of simplification or concentration is continuously adjustable.
- MERS was created with generative tasks in mind, and the consistency that these require.
- The MERS framework is nonparametric and grounded in information theory. Its connections to rate-distortion theory enables us to borrow established tools and techniques from that field.

Presentation Outline

1. Introduction
2. Previous work: Minimum entropy rate simplification (MERS)
    1. Theory
    2. Examples
3. Ongoing work:
    1. MERS for HMMs
    2. Multiplicative-mixture HMMs

MERS for HMMs

In order to use MERS in HMM-based speech synthesis, the ideas need to be extended to finite-duration HMMs.
Since the entropy and divergence rates of HMMs have no known closed-form expressions, we might look into approximate solution techniques from rate-distortion theory for the optimisation.
A simplistic first attempt is to minimise the entropy of the underlying Markov chain and the output distributions separately.

MERS and LISTA

We are thinking about what benefits MERS could bring in a LISTA context. (Ideas are welcome!)

For example, MERS might provide a middle-ground between maximum likelihood parameter generation and simple sampling from the model. MERS enables us to concentrate the output around the most likely behaviour (where the fit of the model is best) to an adjustable degree, while still leaving room for model-consistent variation. This is similar to natural speech but unlike ML output, which is the same every time.

MERS and LISTA

We are thinking about what benefits MERS could bring in a LISTA context. (Ideas are welcome!)
For example, MERS might provide a middle-ground between maximum likelihood parameter generation and simple sampling from the model. MERS enables us to concentrate the output around the most likely behaviour (where the fit of the model is best) to an adjustable degree, while still leaving room for model-consistent variation. This is similar to natural speech but unlike ML output, which is the same every time.

Presentation Outline

1. Introduction
2. Previous work: Minimum entropy rate simplification (MERS)
    1. Theory
    2. Examples
3. Ongoing work:
    1. MERS for HMMs
    2. Multiplicative-mixture HMMs

Perspective on HMMs

We now introduce a currently ongoing line of research at KTH, intended to improve synthesis by developing more appropriate models.

As we all know, HMMs are very popular general sequence models for several reasons:

- They do not require a restrictive Markov assumption for the observations.
- They allow efficient training using the backward-forward procedures and the EM-algorithm.
- They can be set up to use as many (or few) parameters as desired.

Perspective on HMMs

We now introduce a currently ongoing line of research at KTH, intended to improve synthesis by developing more appropriate models.
As we all know, HMMs are very popular general sequence models for several reasons:

- They do not require a restrictive Markov assumption for the observations.
- They allow efficient training using the backward-forward procedures and the EM-algorithm.
- They can be set up to use as many (or few) parameters as desired.

Discrete-State HMMs

The most common HMMs are based on discrete Markov chains with Gaussian or GMM state-conditional observation distributions. Despite their prevalence, these are not good speech models for several reasons:

1. They describe a process as stationary segments separated by instantaneous transitions. Within each epoch, the output is IID around the state-conditional mean. In contrast, speech flows smoothly without such discrete steps.
2. The duration of each sound (or epoch) is exponentially distributed, with the mode at the shortest duration possible. Actual speech sounds are very rarely that short.

In general, natural processes tend to follow a continuous path, like our voice apparatus does. Discrete-state HMMs do not. Because of the important differences, discrete-state HMMs for speech generate highly unnatural output when sampling.

Discrete-State HMMs

The most common HMMs are based on discrete Markov chains with Gaussian or GMM state-conditional observation distributions. Despite their prevalence, these are not good speech models for several reasons:

1. They describe a process as stationary segments separated by instantaneous transitions. Within each epoch, the output is IID around the state-conditional mean. In contrast, speech flows smoothly without such discrete steps.
2. The duration of each sound (or epoch) is exponentially distributed, with the mode at the shortest duration possible. Actual speech sounds are very rarely that short.

In general, natural processes tend to follow a continuous path, like our voice apparatus does. Discrete-state HMMs do not. Because of the important differences, discrete-state HMMs for speech generate highly unnatural output when sampling.

Discrete-State HMMs

The most common HMMs are based on discrete Markov chains with Gaussian or GMM state-conditional observation distributions. Despite their prevalence, these are not good speech models for several reasons:

1. They describe a process as stationary segments separated by instantaneous transitions. Within each epoch, the output is IID around the state-conditional mean. In contrast, speech flows smoothly without such discrete steps.
2. The duration of each sound (or epoch) is exponentially distributed, with the mode at the shortest duration possible. Actual speech sounds are very rarely that short.

In general, natural processes tend to follow a continuous path, like our voice apparatus does. Discrete-state HMMs do not. Because of the important differences, discrete-state HMMs for speech generate highly unnatural output when sampling.

Discrete-State HMMs

The most common HMMs are based on discrete Markov chains with Gaussian or GMM state-conditional observation distributions. Despite their prevalence, these are not good speech models for several reasons:

1. They describe a process as stationary segments separated by instantaneous transitions. Within each epoch, the output is IID around the state-conditional mean. In contrast, speech flows smoothly without such discrete steps.
2. The duration of each sound (or epoch) is exponentially distributed, with the mode at the shortest duration possible. Actual speech sounds are very rarely that short.

In general, natural processes tend to follow a continuous path, like our voice apparatus does. Discrete-state HMMs do not. Because of the important differences, discrete-state HMMs for speech generate highly unnatural output when sampling.

Established Workarounds

To our current understanding, HMM-based synthesis today uses a few tricks to mitigate the discrepancies between model and reality:

1. The issue with quantisation steps (instantaneous transitions) is solved by training delta and delta-delta parameters as well, and incorporating these when computing the maximum likelihood output. This appears mathematically paradoxical—how can an epoch both have constant mean and nonzero expected derivative? What about sampling?
2. The duration problem is addressed either by setting time-in-state equal to the expected time, or by using hidden semi-Markov models. The latter is more elegant, though complex to train.

We here propose a different approach, attacking the problem at the root by proposing an HMM class that better describes smoothly changing processes.

Established Workarounds

To our current understanding, HMM-based synthesis today uses a few tricks to mitigate the discrepancies between model and reality:

1. The issue with quantisation steps (instantaneous transitions) is solved by training delta and delta-delta parameters as well, and incorporating these when computing the maximum likelihood output. This appears mathematically paradoxical—how can an epoch both have constant mean and nonzero expected derivative? What about sampling?
2. The duration problem is addressed either by setting time-in-state equal to the expected time, or by using hidden semi-Markov models. The latter is more elegant, though complex to train.

We here propose a different approach, attacking the problem at the root by proposing an HMM class that better describes smoothly changing processes.

Established Workarounds

To our current understanding, HMM-based synthesis today uses a few tricks to mitigate the discrepancies between model and reality:

1. The issue with quantisation steps (instantaneous transitions) is solved by training delta and delta-delta parameters as well, and incorporating these when computing the maximum likelihood output. This appears mathematically paradoxical—how can an epoch both have constant mean and nonzero expected derivative? What about sampling?
2. The duration problem is addressed either by setting time-in-state equal to the expected time, or by using hidden semi-Markov models. The latter is more elegant, though complex to train.

We here propose a different approach, attacking the problem at the root by proposing an HMM class that better describes smoothly changing processes.

Established Workarounds

To our current understanding, HMM-based synthesis today uses a few tricks to mitigate the discrepancies between model and reality:

1. The issue with quantisation steps (instantaneous transitions) is solved by training delta and delta-delta parameters as well, and incorporating these when computing the maximum likelihood output. This appears mathematically paradoxical—how can an epoch both have constant mean and nonzero expected derivative? What about sampling?
2. The duration problem is addressed either by setting time-in-state equal to the expected time, or by using hidden semi-Markov models. The latter is more elegant, though complex to train.

We here propose a different approach, attacking the problem at the root by proposing an HMM class that better describes smoothly changing processes.

Mixed-State HMMs

First innovation: Discrete-state HMMs are quantised to always be in a single state. We propose an underlying Markov process that moves gradually from one discrete state to the next. At any point in time, the state is then a mixture of the original discrete states.

Since speech is modelled by left-right HMMs, we assume different realisations of a specific sound are just small variations around the same basic path. We thus consider a simple model where the state at t is represented by a position y_t that moves gradually along an axis. Certain points $\left\{y^{(n)}\right\}_{n=1}^{N}$ on the axis are associated with the original, discrete states; the distance to these points determine the mixture composition. Every y-value is associated with a specific set of mixture coefficients $w^{(n)}(y)$. This is a little like radial basis functions.

Mixed-State HMMs

First innovation: Discrete-state HMMs are quantised to always be in a single state. We propose an underlying Markov process that moves gradually from one discrete state to the next. At any point in time, the state is then a mixture of the original discrete states.
Since speech is modelled by left-right HMMs, we assume different realisations of a specific sound are just small variations around the same basic path. We thus consider a simple model where the state at t is represented by a position y_t that moves gradually along an axis.
Certain points $\left\{y^{(n)}\right\}_{n=1}^{N}$ on the axis are associated with the original, discrete states; the distance to these points determine the mixture composition. Every y-value is associated with a specific set of mixture coefficients $w^{(n)}(y)$. This is a little like radial basis functions.

Mixed-State HMMs

First innovation: Discrete-state HMMs are quantised to always be in a single state. We propose an underlying Markov process that moves gradually from one discrete state to the next. At any point in time, the state is then a mixture of the original discrete states.
Since speech is modelled by left-right HMMs, we assume different realisations of a specific sound are just small variations around the same basic path. We thus consider a simple model where the state at t is represented by a position y_t that moves gradually along an axis.
Certain points $\left\{y^{(n)}\right\}_{n=1}^{N}$ on the axis are associated with the original, discrete states; the distance to these points determine the mixture composition. Every y-value is associated with a specific set of mixture coefficients $w^{(n)}(y)$. This is a little like radial basis functions.

State Evolution

We want the state y_t to move gradually along its axis from start to finish. Just like a discrete Markov chain, we can let the current value y_t completely determine the distribution over future y. The HMM formalism then still applies.

For natural durations, let each component be associated with a characteristic movement speed, and a variance around this value. The change in y between frames, $y_{t+1} - y_t$, is then governed by a mixture of the characteristic velocities at y_t, using weights $w^{(n)}$.

This produces variable and non-exponential durations of component sounds. The details are not of crucial importance, only that there is a continuous progress with some jitter on top.

State Evolution

We want the state y_t to move gradually along its axis from start to finish. Just like a discrete Markov chain, we can let the current value y_t completely determine the distribution over future y. The HMM formalism then still applies.

For natural durations, let each component be associated with a characteristic movement speed, and a variance around this value. The change in y between frames, $y_{t+1} - y_t$, is then governed by a mixture of the characteristic velocities at y_t, using weights $w^{(n)}$.

This produces variable and non-exponential durations of component sounds. The details are not of crucial importance, only that there is a continuous progress with some jitter on top.

State Evolution

We want the state y_t to move gradually along its axis from start to finish. Just like a discrete Markov chain, we can let the current value y_t completely determine the distribution over future y. The HMM formalism then still applies.

For natural durations, let each component be associated with a characteristic movement speed, and a variance around this value. The change in y between frames, $y_{t+1} - y_t$, is then governed by a mixture of the characteristic velocities at y_t, using weights $w^{(n)}$.

This produces variable and non-exponential durations of component sounds. The details are not of crucial importance, only that there is a continuous progress with some jitter on top.

Mixture Outputs

In a regular HMM, each state is associated with an output distribution. In the mixed-state scenario, we rather think of these as templates. The progress along the axis determines the current mixture coefficients $w^{(n)}(y_t)$ of the templates; the closer the template $y^{(n)}$ is to y_t, the more like it the output distribution at t will be. Like all HMMs, observations are independent if the corresponding states y are known.

For a simple, first model, we let each template be a Gaussian. These are usually combined in additive mixtures (GMMs)

$$p_{\mathbf{X}}(x) = \sum_n w^{(n)} \frac{1}{(2\pi)^{\frac{M}{2}} \det \boldsymbol{\Sigma}^{(n)}} \exp\left(\left(x - \mu^{(n)}\right)^T \left(\boldsymbol{\Sigma}^{(n)}\right)^{-1} \left(x - \mu^{(n)}\right)\right)$$

However, this does not give the result we desire.

Mixture Outputs

In a regular HMM, each state is associated with an output distribution. In the mixed-state scenario, we rather think of these as templates. The progress along the axis determines the current mixture coefficients $w^{(n)}(y_t)$ of the templates; the closer the template $y^{(n)}$ is to y_t, the more like it the output distribution at t will be. Like all HMMs, observations are independent if the corresponding states y are known.
For a simple, first model, we let each template be a Gaussian. These are usually combined in additive mixtures (GMMs)

$$p_{\boldsymbol{x}}(\boldsymbol{x}) = \sum_n w^{(n)} \frac{1}{(2\pi)^{\frac{M}{2}} \det \boldsymbol{\Sigma}^{(n)}} \exp\left(\left(\boldsymbol{x} - \boldsymbol{\mu}^{(n)}\right)^T \left(\boldsymbol{\Sigma}^{(n)}\right)^{-1} \left(\boldsymbol{x} - \boldsymbol{\mu}^{(n)}\right)\right)$$

However, this does not give the result we desire.

Multiplicative Mixtures

The problem with additive Gaussian mixtures is that, while the components are Gaussian, the output is not. The distribution may be multi-modal, and the mode may not move smoothly between the various template means as weights change. The GMM may also have greater variance than the component distributions. In addition, a mixture of speech sounds may not be a speech sound itself.

Second innovation: We use *multiplicative mixtures* instead. These look like

$$p_{\boldsymbol{X}}(\boldsymbol{x}) \propto \exp\left(\sum_n w^{(n)} \left(\boldsymbol{x} - \boldsymbol{\mu}^{(n)}\right)^T \left(\boldsymbol{\Sigma}^{(n)}\right)^{-1} \left(\boldsymbol{x} - \boldsymbol{\mu}^{(n)}\right)\right).$$

The output is always Gaussian, with a mode and variance that smoothly intermediates the extremes defined by the templates.
This appears to be a novel idea for HMMs.

Multiplicative Mixtures

The problem with additive Gaussian mixtures is that, while the components are Gaussian, the output is not. The distribution may be multi-modal, and the mode may not move smoothly between the various template means as weights change. The GMM may also have greater variance than the component distributions. In addition, a mixture of speech sounds may not be a speech sound itself.

Second innovation: We use *multiplicative mixtures* instead. These look like

$$p_{\boldsymbol{X}}(\boldsymbol{x}) \propto \exp\left(\sum_n w^{(n)} \left(\boldsymbol{x}-\boldsymbol{\mu}^{(n)}\right)^T \left(\boldsymbol{\Sigma}^{(n)}\right)^{-1} \left(\boldsymbol{x}-\boldsymbol{\mu}^{(n)}\right)\right).$$

The output is always Gaussian, with a mode and variance that smoothly intermediates the extremes defined by the templates.

This appears to be a novel idea for HMMs.

Practical Considerations

1. A proposed HMM model is only useful if it can be trained efficiently in practise. In our first attempts, we restricted ourselves to diagonal covariance matrices $\mathbf{\Sigma}^{(n)}$ and considered the template centres $y^{(n)}$ fixed. Preliminary derivations indicate that movement speed parameters and template parameters can be updated independently by solving a number of linear systems.
2. EM-algorithm parameter updates require the hidden state probabilities $P(S_t = s \mid \underline{\boldsymbol{X}})$, computed using the forward-backward algorithm. For discrete S_t, these can be exhaustively described by a matrix. However, $P(Y_t = y \mid \underline{\boldsymbol{X}})$ is a distribution over a line segment, and may not be described by a finite set of numbers. This can be circumvented by simply discretising (quantising) the y-line to form an ordinary Markov chain, with a transition matrix that approximates the y-motion. Quantisation resolution controls approximation error.

Practical Considerations

1. A proposed HMM model is only useful if it can be trained efficiently in practise. In our first attempts, we restricted ourselves to diagonal covariance matrices $\mathbf{\Sigma}^{(n)}$ and considered the template centres $y^{(n)}$ fixed. Preliminary derivations indicate that movement speed parameters and template parameters can be updated independently by solving a number of linear systems.
2. EM-algorithm parameter updates require the hidden state probabilities $P(S_t = s \mid \underline{\boldsymbol{X}})$, computed using the forward-backward algorithm. For discrete S_t, these can be exhaustively described by a matrix. However, $P(Y_t = y \mid \underline{\boldsymbol{X}})$ is a distribution over a line segment, and may not be described by a finite set of numbers. This can be circumvented by simply discretising (quantising) the y-line to form an ordinary Markov chain, with a transition matrix that approximates the y-motion. Quantisation resolution controls approximation error.

Relevance to LISTA

The proposed multiplicative-mixture HMMs should require fewer parameters than traditional approaches. Consequently, they may need less data to train and adapt. This is highly interesting within LISTA.

There are two main reasons for this efficiency:

1. Gradual transitions can be achieved without including and training parameters relating to delta and delta-delta coefficients. Smoothness is accomplished in a mathematically consistent manner.
2. Because the output is naturally smooth, the need for additional states to represent intermediary sounds is reduced.

Generally, we may expect better results because our models are set up to fit human speech better and easier.

Our first models include two motion parameters to be adapted per state $y^{(n)}$ of the Markov process (same as many HSMMs) plus Gaussian parameters for each template.

Relevance to LISTA

The proposed multiplicative-mixture HMMs should require fewer parameters than traditional approaches. Consequently, they may need less data to train and adapt. This is highly interesting within LISTA.
There are two main reasons for this efficiency:

1. Gradual transitions can be achieved without including and training parameters relating to delta and delta-delta coefficients. Smoothness is accomplished in a mathematically consistent manner.
2. Because the output is naturally smooth, the need for additional states to represent intermediary sounds is reduced.

Generally, we may expect better results because our models are set up to fit human speech better and easier.
Our first models include two motion parameters to be adapted per state $y^{(n)}$ of the Markov process (same as many HSMMs) plus Gaussian parameters for each template.

Relevance to LISTA

The proposed multiplicative-mixture HMMs should require fewer parameters than traditional approaches. Consequently, they may need less data to train and adapt. This is highly interesting within LISTA.
There are two main reasons for this efficiency:

1. Gradual transitions can be achieved without including and training parameters relating to delta and delta-delta coefficients. Smoothness is accomplished in a mathematically consistent manner.
2. Because the output is naturally smooth, the need for additional states to represent intermediary sounds is reduced.

Generally, we may expect better results because our models are set up to fit human speech better and easier.

Our first models include two motion parameters to be adapted per state $y^{(n)}$ of the Markov process (same as many HSMMs) plus Gaussian parameters for each template.

Relevance to LISTA

The proposed multiplicative-mixture HMMs should require fewer parameters than traditional approaches. Consequently, they may need less data to train and adapt. This is highly interesting within LISTA.
There are two main reasons for this efficiency:

1. Gradual transitions can be achieved without including and training parameters relating to delta and delta-delta coefficients. Smoothness is accomplished in a mathematically consistent manner.
2. Because the output is naturally smooth, the need for additional states to represent intermediary sounds is reduced.

Generally, we may expect better results because our models are set up to fit human speech better and easier.

Our first models include two motion parameters to be adapted per state $y^{(n)}$ of the Markov process (same as many HSMMs) plus Gaussian parameters for each template.

Relevance to LISTA

The proposed multiplicative-mixture HMMs should require fewer parameters than traditional approaches. Consequently, they may need less data to train and adapt. This is highly interesting within LISTA.
There are two main reasons for this efficiency:

1. Gradual transitions can be achieved without including and training parameters relating to delta and delta-delta coefficients. Smoothness is accomplished in a mathematically consistent manner.
2. Because the output is naturally smooth, the need for additional states to represent intermediary sounds is reduced.

Generally, we may expect better results because our models are set up to fit human speech better and easier.
Our first models include two motion parameters to be adapted per state $y^{(n)}$ of the Markov process (same as many HSMMs) plus Gaussian parameters for each template.

Summary

- We introduced a class of HMM models intended to describe naturally smooth and continuous processes such as speech.
- The two main innovations involved are the definition of an underlying Markov process that moves seamlessly between states, along with using multiplicative-mixture output distributions.
- We expect these to produce better speech models with less data, making them interesting for LISTA. Specifically, these models may need fewer parameters, and should thus adapt faster.
- Efficient implementations of training and sampling, similar to regular discrete-state HMMs, appear likely.
- While we have no concrete results yet, this seems like an innovative and promising approach.

That's All, Folks!

That's All, Folks!

Thank you for listening!